Scales

Amina tem seis moedas, numeradas de 1 a 6 e sabe que as moedas têm todas pesos diferentes. Ela gostaria de ordená-las de acordo com os seus pesos. Para tanto, Amina desenvolveu um novo tipo de balança.

Uma balança tradicional tem dois pratos. Para usar uma dessas balanças, você coloca uma moeda em cada prato e determina qual das moedas é a mais pesada.

A nova balança da Amina é mais complexa. Tem quatro pratos, rotulados A, B, C e D. A balança tem quatro configurações diferentes, cada uma das quais responde a uma pergunta diferente em relação às moedas. Para usar a balança, é necessário colocar exatamente uma moeda em cada um dos pratos A, B e C. Além disso, na quarta configuração, é necessário também colocar exatamente uma moeda no prato D.

As quatro configurações fazem a balança responder às quatro perguntas abaixo:

1. Qual das moedas nos pratos A, B e C é a mais pesada?
2. Qual das moedas nos pratos A, B e C é a mais leve?
3. Qual das moedas nos pratos A, B e C é a mediana? (Esta é a moeda que não é nem a mais pesada nem a mais leve das três.)
4. Das moedas nos pratos A, B e C, considere apenas as moedas que são mais pesadas do que a moeda no prato D. Se existem moedas assim, qual é a mais leve? Por outro lado, se não existem moedas assim, qual das moedas nos pratos A, B e C é a mais leve?

Tarefa

Escreva um programa que ordena as seis moedas de Amina de acordo com o seu peso. O programa pode consultar a balança para comparar os pesos das moedas. Seu programa receberá vários casos de teste para resolver, cada um correspondendo a um novo conjunto de seis moedas.

Seu programa deve implementar as funções `init` e `orderCoins`. A cada execução do seu programa, o avaliador chamará `init` precisamente uma vez. Isto fornecerá o número de casos de teste e lhe permitirá inicializar quaisquer variáveis. O avaliador então chamará `orderCoins()` uma vez para cada caso de teste.

- `init(T)`
  - `T`: O número de casos de teste que seu programa terá que resolver nesta execução. `T` é um inteiro no intervalo $1, \ldots, 18$.
  - Esta função não retorna nenhum valor.
- `orderCoins()`

- Esta função é chamada exatamente uma vez para cada caso de teste.
- A função deve determinar a ordem correta das moedas da Amina, chamando as funções `getHeaviest()`, `getLightest()`, `getMedian()`, e/ou `getNextLightest()` do avaliador.
- Uma vez que a função determina a ordem correta, deve reportá-la, chamando a função `answer()` do avaliador.
- Depois de chamar `answer()`, a função `orderCoins()` deve retornar, sem devolver nenhum valor.

No seu programa, você pode usar as seguintes funções do avaliador:

- `answer(W)` — seu programa deve usar esta função para indicar a resposta que determinou.
  - `W`: Um vetor de comprimento 6 que contém a ordem correta das moedas. `W[0]` a `W[5]` devem ser os números das moedas (i.e., números de 1 a 6) na ordem, da mais leve para a mais pesada.
  - Seu programa deve somente chamar esta função a partir de `orderCoins()`, uma vez para cada caso de teste.
  - Esta função não retorna nenhum valor.
- `getHeaviest(A, B, C)`, `getLightest(A, B, C)`, `getMedian(A, B, C)` — estas funções correspondem, respectivamente, às configurações 1, 2 e 3 da balança da Amina.
  - `A`, `B`, `C`: As moedas são colocadas nos pratos A, B e C, respectivamente. `A`, `B` e `C` devem ser três inteiros distintos, todos no interrvalo entre 1 e 6 inclusive.
  - Cada função retorna um dos números `A`, `B`, e `C`: o número da moeda apropriada. Por exemplo, `getHeaviest(A, B, C)` retorna o número da moeda mais pesada das três.
- `getNextLightest(A, B, C, D)` — esta corresponde à configuração 4 da balança de Amina.
  - `A`, `B`, `C`, `D`: As moedas que são colocadas nos pratos A, B, C e D, respectivamente. `A`, `B`, `C` e `D` devem ser quatro inteiros distintos, todos no intervalo de 1 a 6, inclusive.
  - A função retorna um dos números `A`, `B` e `C`: o número da moeda selecionada pela balança, como descrito acima, na configuração 4. Assim, a moeda retornada é a mais leve dentre as moedas nos pratos A, B e C que são mais pesadas que a moeda no prato D, ou, se nenhuma delas for mais pesada do que a moeda no prato D, a moeda retornada é simplesmente a mais leve das três moedas nos pratos A, B e C.

Pontuação

Não há subtarefas neste problema. A pontuação será baseada em quantas pesagens o seu programa executa (número total de chamadas para as funções `getLightest()`, `getHeaviest()`, `getMedian()` e/ou `getNextLightest()`).

Seu programa será executado várias vezes, cada vez com múltiplos casos de teste. Seja r o número de execuções do seu programa. Este número está fixado pelos dados de teste. Se seu programa não ordenar as moedas corretamente em qualquer caso de teste de qualquer execução, receberá 0 pontos.

Caso contrário, as execuções serão pontuados individualmente, como segue abaixo.

Seja Q o menor número de pesagens para ordenar qualquer sequência de seis moedas usando a balança de Amina. Para tornar a tarefa mais desafiadora, não vamos revelar o valor de Q agora.

Suponha que o maior número de pesagens entre todos os casos de teste de todas as execuções seja $Q + y$, para algum inteiro y. Então, considere uma execução em particular do seu programa. Seja $Q + x$ o maior número de pesagens desta execução, considerando todos os testes de caso, onde x é um número não negativo. (Se você usar menos do que Q pesagens para todos os casos de teste desta execução, então $x = 0$.) Então, a pontuação desta execução será $\frac{100}{r((x+y)/5+1)}$, arrendondado *para baixo* com precisão de dois algarismos depois da vírgula.

Em particular, se seu programa fizer no máximo Q pesagens para cada caso teste de cada execução, você receberá 100 pontos.

Exemplo

Suponha que a ordem das moedas seja $3\ 4\ 6\ 2\ 1\ 5$, da mais leve para a mais pesada.

Função chamada	Retorna	Explicação
getMedian(4, 5, 6)	6	A moeda 6 é a mediana entre as moedas 4, 5 e 6.
getHeaviest(3, 1, 2)	1	A moeda 1 é a mais pesada entre as moedas 1, 2 e 3.
getNextLightest(2, 3, 4, 5)	3	As moedas 2, 3 e 4 são todas mais leves do que a moeda 5, então a mais leve entre elas (3) é retornada.
getNextLightest(1, 6, 3, 4)	6	As moedas 1 e 6 são ambas mais pesadas do que a moeda 4. Entre as moedas 1 e 6, a moeda 6 é a mais leve.
getHeaviest(3, 5, 6)	5	A moeda 5 é a mais pesada entre as moedas 3, 5 e 6.
getMedian(1, 5, 6)	1	A moeda 1 é a mediana entre as moedas 1, 5 e 6.
getMedian(2, 4, 6)	6	A moeda 6 é a mediana entre as moedas 2, 4 e 6.
answer([3, 4, 6, 2, 1, 5])		O programa determinou a resposta correta para este caso de teste.

Avaliador exemplo

O avaliador exemplo lê dados no seguinte formato:

- linha 1: T —- o número de casos de teste
- cada uma das linhas de 2 a $T + 1$: uma sequência de 6 números distintos de 1 a 6: a ordem das moedas, da mais leve à mais pesada.

Por exemplo, se tivermos dois casos de teste, em que as moedas são ordenadas respectivamente $1\ 2\ 3\ 4\ 5\ 6$ e $3\ 4\ 6\ 2\ 1\ 5$, a entrada será a seguinte:

```
2
1 2 3 4 5 6
3 4 6 2 1 5
```

O avaliador exemplo imprime o vetor que foi passado como parâmetro para a função `answer()`.